Trabalho 4 - Processamento de Imagens

Décio Gonçalves de Aguiar Neto[1]

[1]Universidade Estadual de Campinas - UNICAMP
Instituto de Computação

1. INTRODUÇÃO

Este trabalho tem como objetivo apresentar o uso de técnicas de interpolação aplicadas as operações de rotação e escala, os métodos aqui implementados buscam fazer o preenchimento das imagens de saída através de um mapeamento de pixeis da imagem de saída para a imagem de entrada, a forma como esses pontos são interpolados interferem na qualidade da imagem e no custo computacional.

Métodos de interpolação mais simples, fazem uso de menos pontos na imagem de entrada para obter o pixel desejado na imagem de saída. Métodos mais robustos tendem a usar vizinhanças maiores, assim o pixel a ser preenchido recebe alguma forma de contribuição do valor de pixeis adjacentes na composição do seu nível de cinza.

Os métodos de interpolação implementados neste trabalho foram, o método do vizinho mais próximo, interpolação bilinear, bi-cubica e a interpolação por polinômio de Lagrange. A Seção 2 faz um resumo do funcionamento dos métodos de interpolação implementados e os detalhes que precisaram ser tratados durante sua implementação, na Seção 2.1 são descritos os métodos de escala e rotação e como as interpolações foram utilizadas para implementar estas transformações geométricas.

2. Métodos

Nesta seção são descritos os métodos implementados neste trabalho, começando pelas transformações geométricas e após uma breve explicação de cada uma dos métodos de interpolação que foram implementados para trabalhar junto as transformações geométricas.

2.1. Transformações Geométricas

Nesta seção são apresentadas as transformações geométricas utilizadas neste trabalho e a forma a qual foram implementadas e algumas observações que trouxeram melhorias visuais a aplicação, como será melhor descrito no caso da rotação.

2.1.1. Escala

A transformação de escala aplicada a imagens, consiste em, dada uma imagem de entrada com dimensões H e W, queremos alterar esses valores de forma a não perder as informações contidas na imagem original. Seja uma imagem de entrada com altura H e largura W, e quisermos escalar essa imagem em um valor S, a nova imagem deverá ter dimensões S*H e S*W, para $S > 0$.

Uma outra forma de ver a escala de uma imagem é que invés de passarmos um único valor de escala pelos quais a altura e largura serão multiplicados, podemos dar de

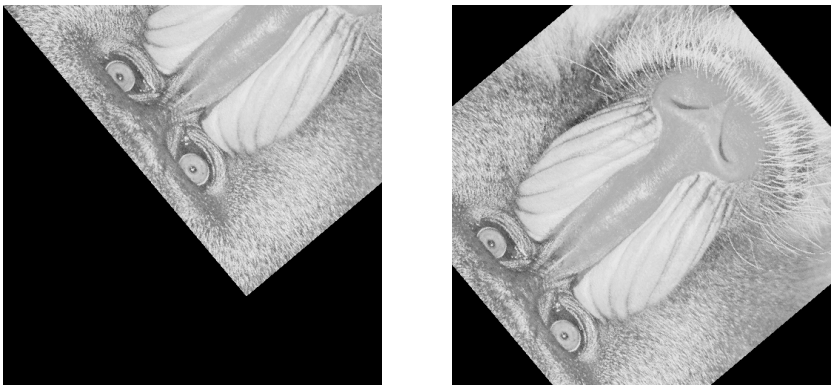
entrada os valores de altura e largura que desejamos que a nova imagem tenha, assim teremos duas escalas, uma escala para a altura e outra para a largura. A Equação 1 descreve como é feito o cálculo das novas escalas, sendo a entrada H' e W' temos agora Sh e Sw como as novas escalas para cada dimensão, e assim podemos fazer o preenchimento da imagem de saída.

$$\begin{aligned} Sh &= \frac{H'}{H} \\ Sw &= \frac{W'}{W} \end{aligned} \tag{1}$$

2.1.2. Rotação

O método de rotação, consiste na aplicação da matriz de rotação sobre os pontos da imagem a fim de obter o mapeamento entre a imagem de entrada e saída, semelhante ao que ocorre na escala faremos uso dos métodos de interpolação para fazer o preenchimento dos pixeis da imagem de saída após o mapeamento. Algumas informações são interessantes de serem destacadas nesta transformação, se aplicarmos apenas a matriz de rotação para obter o mapeamento, teremos caso que perderemos as informações da imagem de entrada, a rotação acontecerá mas como o eixo global da moldura se encontra no canto superior esquerdo, fazemos a rotação em relação a esse eixo, enquanto a imagem possui seu próprio eixo local, diferente da origem, o resultado é que a imagem não ficará inteiramente na moldura como mostrado na Figura 2.1.2.

Para corrigirmos e obtermos um resultado visível, aplicamos uma translação no ponto a ser mapeado de forma que o mesmo seja posto na origem, aplicamos a rotação e após aplicamos a translação novamente de forma a voltar para o eixo local e seja feita a interpolação do pixel, o resultado dessa operação pode ser observado na Sub-Figura 2.1.2, diferente da Sub-Figura 2.1.2 conseguimos rotacionar a imagem de forma que não temos grandes percas de informações.



(a) Imagem sem Translação (b) Imagem com Translação

Figura 1. Imagens lado a lado

2.2. Interpolações

As interpolações aqui implementadas tem como objetivo analisar as diferentes complexidades dessa tarefa, desde uma interpolação mais simples como a de vizinho mais próximo que se utiliza apenas de um único ponto como amostra para o preenchimento do ponto da imagem de saída, a interpolação complexa como a de polinômio de Lagrange que usa uma grande vizinhança para obter o valor de intensidade do pixel a ser preenchido.

2.2.1. Vizinho Mais Próximo

Na interpolação por vizinho mais próximo dado um ponto de entrada F(x',y'), queremos saber qual intensidade da imagem original irá preencher este ponto da imagem de saída, para isso iremos analisar somente o intervalo onde se encontram seu dx e dy descritos pela Equação 2. Esses valores se encontram no intervalo $0 <= dx <= dy <= 1$, e o pixel de saída será preenchido por um dos 4 pixeis da vizinhança, no caso na direção de dx e dy.

$$\begin{aligned} dx &= x' - \lfloor x' \rfloor \\ dy &= y' - \lfloor y' \rfloor \end{aligned} \tag{2}$$

2.2.2. Interpolação Bilinear

Nesta interpolação diferente da vizinha mais próximo que usa o valor de apenas um pixel para preencher o pixel da imagem de saída, nesta utilizamos uma espécia de média ponderada de uma vizinhança de 4 pixels, assim fazendo com que exista contribuição de pixeis diferentes de forma que o preenchimento do pixel da imagem de saída seja mais completo que o método anterior.

2.2.3. Bi-Cubica

A interpolação Bi-Cubica, demanda um maior custo computacional, neste método fazemos uso de 16 pixeis da vizinhança para estimar o nível de cinza que está sendo mapeado, uma informação importante neste método é como as bordas serão tratadas já que quando chegamos nos extremos da imagem parte dos pixeis necessários para a vizinhança não irão existir, para isso a decisão de implementação utilizada foi ignorar os valores que não estava dentro da imagem, ou seja no pixel (0,0) por exemplo invés de serem usados 16 vizinhos são usados apenas 9 pixels.

2.2.4. Lagrange

Parecido com a Bi-Cubica, o método de Lagrange obtém bons resultados, mas demanda maior complexidade computacional por também utilizar uma vizinhança maior de pixels, a decisão de implementação para esse método também foi ignorar pixeis fora da borda semelhante ao que foi feito na BI-Cubica.

3. EXECUÇÃO

Para que o código seja executado de forma correta basta executar os seguintes comandos:

1. -i caminho da imagem de entrada
2. -a angulo a ser rotacionado em graus
3. -e fator de escala
4. -m modo de interpolação
    - 0 - vizinho mais próximo
    - 1 - bilinear
    - 2 - bi-cubica
    - 3 - polinômio de Lagrange
5. -d altura largura
6. -o caminho da imagem de saída ex: out.png

Exemplo de entrada:

```
python3 T4.py -i entrada.png -e 2.0 -m 2 -o out.png
```

4. RESULTADOS

Nesta seção são exibidos resultados comparativos dos métodos de interpolação utilizados. a Figura 4 exibe os resultados obtidos utilizando cada um dos métodos de interpolação implementados ao realizar uma escala de 2.0.

Na Sub-Figura 4.2.1 foi observado pouca qualidade quanto ao preenchimento dos pontos, sendo possível observar certas descontinuidades. Na Sub-Figura 4.2.2 foi observado que houve uma certa suavização em relação a imagem original, a aplicação dessa interpolação causou um efeito semelhante a filtros passa baixa. Na Sub-Figura 4.2.3 obtivemos um bom resultado, sem percas visíveis de informação da imagem original o mesmo aconteceu com o uso da interpolação de Lagrange como pode ser observado na Sub-Figura 4.2.4.

(a) Vizinho Mais Proximo

(b) Bi-cubica

(c) Bilinear

(d) Lagrange

Figura 2. Comparativo entre os resultados obtidos em cada interpolação